Amendments to the Claims

This following listing of claims will replace all prior versions and listings of claims in the application.

1. (currently amended) A method of representing a subject register in an emulator ~~(20)~~, the method comprising the steps of:

(a) mapping an abstract register representing a subject register of a subject machine to either a first location or to a second location within a target machine; and

(b) alternating mapping of the abstract register between the first and second locations such that ~~the content of~~ one of the first or second locations represents a definitive version of the abstract register for use by the emulator during exception handling, whilst the other of the first or second locations represents a speculative version of the abstract register.

2. (original) A method as claimed in claim 1, wherein for a predetermined section of subject code, one of the first or second locations holds a definitive value of the abstract register at entry into that section, whilst the other of the first or second locations holds a speculative current value of the abstract register for use during that section.

3. (currently amended) A method as claimed in claim 2, wherein the step ~~(b)~~ of alternating mapping is performed upon reaching the end of the predetermined section of subject code.

4. (currently amended) A method as claimed in claim 3, wherein the step ~~(b)~~ of alternating mapping is performed only if the content of the speculative version of the abstract register ~~(X)~~ has been updated during the predetermined section of subject code.

5. (currently amended). A method as claimed in claim 1, further comprising ~~wherein the step (a) comprises the steps of~~:

~~(a1)~~ providing a plurality of abstract registers each representing a register of the subject machine; and

~~(a2)~~ mapping each of the plurality of abstract registers to either a respective one of a first set of locations or a respective one of a second set of locations within the target machine; and

~~wherein the step (b) comprises~~ alternating mapping for each of the abstract registers between the respective one of each of the first and second sets of locations.

6. (original) A method as claimed in claim 1, wherein the first location and the second location are each a memory location or a target register.

7. (original) A method as claimed in claim 2, wherein the predetermined section of subject code represents one or more basic blocks of subject code.

8. (original) A method as claimed in claim 2, wherein the method is for use in an emulator that performs dynamic binary translation.

9. (original) A method for use in handling exceptions by an emulator performing program code conversion between subject code suitable for a subject processor and target code suitable for a target processor, the method comprising the steps of:

(a) providing at least one abstract register each representing a register of the subject processor;

(b) mapping the or each abstract register to a corresponding pair of locations within the

target processor; and

(c) alternating mapping of the or each abstract register between a first of the pair of locations and a second of the pair of locations, such that for a predetermined section of subject code, one of the first or second locations holds a definitive value of the abstract register at entry into that section for use by the emulator during exception handling, whilst the other of the first or second locations holds a speculative current value of the abstract register for updating by the emulator during that section.

10. (currently amended) A method as claimed in claim 9, wherein the step ~~(C)~~ of alternating mapping is performed upon reaching the end of the predetermined section of subject code.

11. (currently amended) A method as claimed in claim 10, wherein the step ~~(C)~~ of alternating mapping is performed only for the or each abstract register that has been updated during the predetermined section of subject code.

12. (original) An emulator apparatus, comprising:

an emulator for mapping an abstract register representing a subject register of a subject machine to either a first location or to a second location within a target machine, the emulator for alternating mapping of the abstract register between the first and second locations such that the content of one of the first or second locations represents a definitive version of the abstract register for use by the emulator during exception handling, whilst the other of the first or second locations represents a speculative version of the abstract register.

13 (original). An emulator apparatus for performing program code conversion between subject code suitable for a subject processor and target codes suitable for a target processor, the emulator

apparatus comprising:

at least one abstract register each representing a register of the subject processor; and

an emulator for mapping the or each abstract register to a corresponding pair of locations within the target processor, and for alternating mapping of the or each abstract register between a first of the pair of locations and a second of the pair of locations, such that for a predetermined section of subject code, one of the first or second locations holds a definitive value of the abstract register at entry into that section for use by the emulator during exception handling, whilst the other of the first or second locations holds a speculative current value of the abstract register for updating the emulator during that section.

14 (currently amended). A computer program product containing computer readable instructions for performing a method of representing a subject register in an emulator, the method comprising the steps of:

(a) mapping an abstract register representing a subject register of a subject machine to either a first location or to a second location within a target machine; and

(b) alternating mapping of the abstract register between the first and second locations such that ~~the content of~~ one of the first or second locations represents a definitive version of the abstract register for use by the emulator during exception handling, whilst the other of the first or second locations represents a speculative version of the abstract register.

15 (original). A computer program product containing computer readable instructions for performing program code conversion between subject code suitable for a subject processor and target codes suitable for a target processor, the method comprising the steps of:

(a) providing at least one abstract register each representing a register of the subject processor;

(b) mapping the or each abstract register to a corresponding pair of locations within the target processor; and

(c) alternating mapping of the or each abstract register between a first of the pair of locations and a second of the pair of locations, such that for a predetermined section of subject code, one of the first or second locations holds a definitive value of the abstract register at entry into that section for use by the emulator during exception handling, whilst the other of the first or second locations holds a speculative current value of the abstract register for updating by the emulator during that section.